NSA BACKEND

CODE ANALYSIS

By: Gateway

2022-03-29

CONTENT

INTRODUCTION

This document contains results of the code analysis of NSA Backend.

CONFIGURATION

- Quality Profiles
  - Names: Sonar JavaScript Alternate [JavaScript];
  - Files: AX-yB5hZsENHbFRYVuf4.json;
- Quality Gate
  - Name: Stryker Backend
  - File: Stryker Backend.xml

SYNTHESIS

ANALYSIS STATUS

Reliability	Security	Security Review	Maintainability
A	A	A	A

QUALITY GATE STATUS

Quality Gate Status	Passed

Metric	Value
Reliability Rating	OK
Security Rating	OK
Maintainability Rating	OK
Security Hotspots Reviewed	OK

METRICS

Coverage	Duplication	Comment density	Median number of lines of code per file	Adherence to coding standard
0.0 %	28.4 %	18.5 %	243.0	99.9 %

TESTS

Total	Success Rate	Skipped	Errors	Failures
0	0 %	0	0	0

DETAILED TECHNICAL DEBT

Reliability	Security	Maintainability	Total
-	-	0d 0h 30min	**0d 0h 30min**

METRICS RANGE

	Cyclomatic Complexity	Cognitive Complexity	Lines of code per file	Comment density (%)	Coverage	Duplication (%)
Min	0.0	0.0	0.0	0.0	0.0	0.0
Max	2987.0	5912.0	15965.0	100.0	0.0	98.9

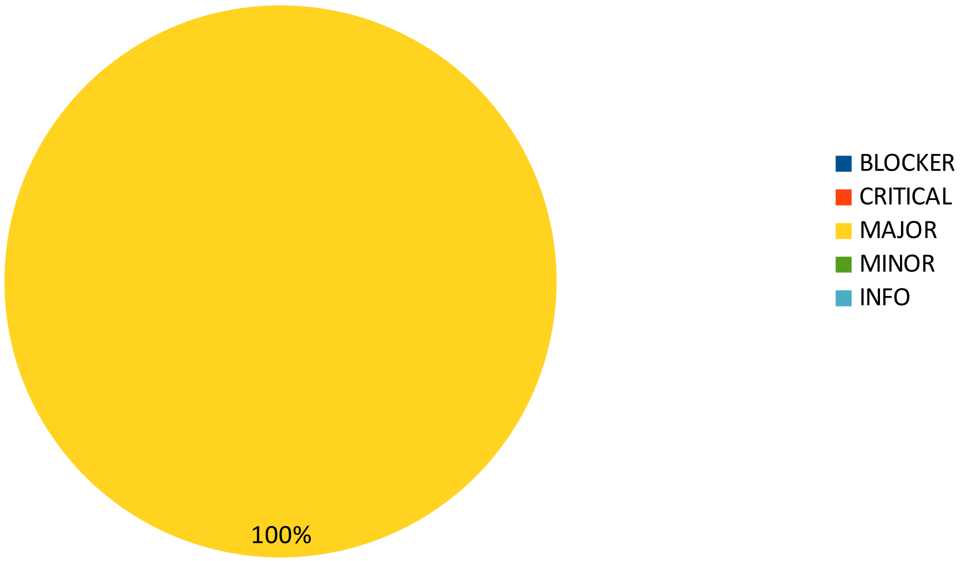
VOLUME

Language	Number
JavaScript	20979
Total	20979

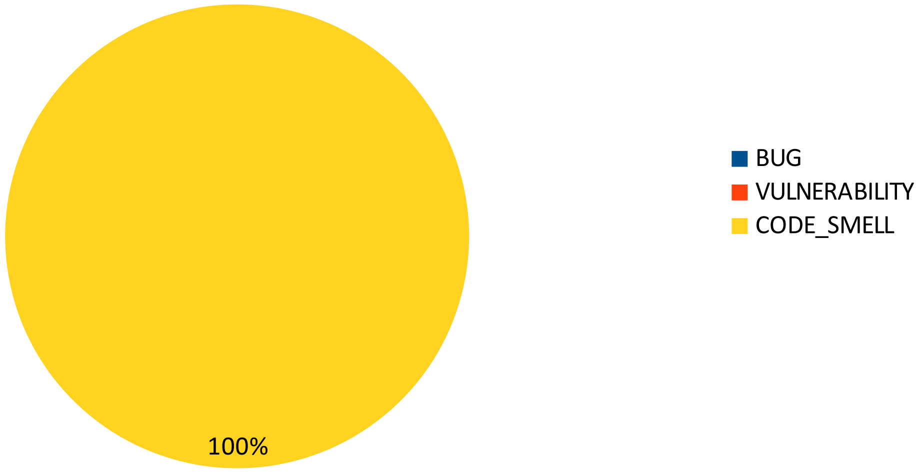
ISSUES

CHARTS

Number of issues by severity



Number of issues by type



ISSUES COUNT BY SEVERITY AND TYPE

Type / Severity	INFO	MINOR	MAJOR	CRITICAL	BLOCKER
BUG	0	0	0	0	0
VULNERABILITY	0	0	0	0	0
CODE_SMELL	0	0	1	0	0

ISSUES LIST

Name	Description	Type	Severity	Number
Functions should not be defined inside loops	Defining a function inside of a loop can yield unexpected results. Such a function keeps references to the variables which are defined in outer scopes. All function instances created inside the loop therefore see the same values for these variables, which is probably not expected. Noncompliant Code Example var funs =	CODE_SMELL	MAJOR	1

	[]; for (var i = 0; i &lt; 13; i++) { funs[i] = function() { // Non-Compliant return i; }; } console.log(funs[0] ()); // 13 instead of 0 console.log(funs[1] ()); // 13 instead of 1 console.log(funs[2] ()); // 13 instead of 2 console.log(funs[3] ()); // 13 instead of 3 ...	

SECURITY HOTSPOTS

SECURITY HOTSPOTS COUNT BY CATEGORY AND PRIORITY

Category / Priority	LOW	MEDIUM	HIGH
LDAP Injection	0	0	0
Object Injection	0	0	0
Server-Side Request Forgery (SSRF)	0	0	0
XML External Entity (XXE)	0	0	0
Insecure Configuration	0	0	0
XPath Injection	0	0	0
Authentication	0	0	0
Weak Cryptography	0	0	0
Denial of Service (DoS)	0	0	0
Log Injection	0	0	0
Cross-Site Request Forgery (CSRF)	0	0	0
Open Redirect	0	0	0